REMARKS

Claims 1-18 are now pending in the application. The following remarks are believed to be fully responsive to the outstanding Office Action and are believed to place the application in condition for allowance. The Examiner is respectfully requested to reconsider and withdraw the rejections in view of the remarks contained herein.

REJECTION UNDER 35 U.S.C. § 103

Claims 1, 3-7, 9-13, and 15-18 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over "Microsoft Device Driver for Symbios Logic ATA/ATAPI-to-1394 Controller Included in Microsoft's New NT5 Beta DDK Release, 10/6/1997" (hereinafter referred to as Microsoft '997) in view of Harris et al. (U.S. Pub. No. 2002/0081873).

Claims 2, 8, and 14 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Microsoft '997 in view of Harris as applied to Claim 7 above, and further in view of Hatano (U.S. Pub. No. 2002/0002645).

These rejections are respectfully traversed.

Claim 7 requires "an interface arranged to convert commands received from the command bus in a format in accordance with one of the ATA/IDE standard and the Serial ATA standard into a format in accordance with said one of the IEEE 1394 standard and the Universal Serial Bus standard[.]" Further, claim 7 recites "[the interface arranged] to supply the converted commands to the terminals for connection to the external databus."

As argued in the previous response, the SYM13FW500 controller shown in Microsoft '997 converts commands in the opposite direction to that required by claim 7. See e.g. pages 12-13 of the response submitted on January 27, 2010.

In the outstanding final Office action, the Examiner asserts that "the 1394-to-ATAPI bridge is intended to convert commands and to supply the comments to both the device and the host apparatus. Further, Examiner respectfully submits that one of ordinary skill in the art could provide for command conversion in the opposite direction as necessary." See page 10 of the outstanding Office action. Applicant respectfully disagrees with the Examiner.

Applicant hereby submits four references to show that the Examiner has made technical errors. The four references are listed at the end of this paper. From the submitted references, it is clear that in IEEE 1394 or USB commands can be only transmitted unidirectionally from the host apparatus to the storage medium device.

Accordingly, the SYM13FW500 controller shown in Microsoft '997 cannot be arranged in a host apparatus as the Examiner alleges. The SYM13FW500 controller converts commands in the opposite direction to that required by claim 7. This is explicitly stated in Microsoft '997 as follows: "The SYM13FW500 controller accepts native 1394 commands and translates them to ATA/ATAPI commands." Microsoft '997, p. 1. There is no disclosure in Microsoft '997 of a command being converted in the direction required by claim 7, i.e., from ATA/IDE standard or Serial ATA standard into IEEE 1394 standard or USB standard, because the SYM13FW500 controller is in the storage medium device on the downstream side of the external databus and only

converts commands (from 1394 standard to ATA) in the opposite direction to that required by claim 7.

More specifically, the submitted references show that in each alternative, commands are transmitted unidirectionally from the host apparatus to the storage medium device, even though data stored on the storage medium device is transmitted bidirectionally.

USB standard

Reference 1 [USBCOMP], parts a and b are extracts from a book concerning the USB standard, and Reference 2 [USB20] is an extract from the specification of the USB standard, version 2.0.

Reference 1 [USBCOMP], part a states:

> Each transaction begins when the host sends a block of information that includes the address of the receiving device and a specific location, called an endpoint, within the device. Everything a device sends is in response to receiving a packet sent by the host.

This "block of information" is a command transmitted unidirectionally from the host apparatus to the storage medium device and is explained in more detail in Reference 2 [USB20], which states:

> 8.5.2 Bulk Transactions
> Bulk transaction types are characterized by the ability to guarantee error-free delivery of data between the host and a function by means of error detection and retry. Bulk transactions use a three-phase transaction consisting of token, data, and handshake packets as shown in Figure 8-30. Under certain flow control and halt conditions, the data phase may be replaced with a handshake resulting in a two-phase transaction in which no data is transmitted. The PING and NYET packets must only be used with devices operating at high-speed.

> When the host is ready to receive bulk data, it issues an IN token. The function endpoint responds by returning either a data packet or, should it be unable to return data, a NAK or STALL handshake. NAK indicates that the function is temporarily unable to return data, while STALL indicates that the endpoint is permanently halted and requires USB System Software intervention. If the host receives a valid data packet, it responds with an ACK handshake. If the host detects an error while receiving data, it returns no handshake packet to the function.
>
> When the host is ready to transmit bulk data, it first issues an OUT token packet followed by a data packet (or PING special token packet, see Section 8.5.1). If the data is received without error by the function, it will return one of three (or four including NYET, for a device operating at high-speed) handshakes:
> • ACK indicates that the data packet was received without errors and informs the host that it may send the next packet in the sequence.
> • NAK indicates that the data was received without error but that the host should resend the data because the function was in a temporary condition preventing it from accepting the data (e.g., buffer full).
> • If the endpoint was halted, STALL is returned to indicate that the host should not retry the transmission because there is an error condition on the function.
> If the data packet was received with a CRC or bit stuff error, no handshake is returned.
>
> Thus whether the host is requesting data from the device or sending data to the device, the data transfer always begins with the host issuing an In or Out token to the device function endpoint. The device will return data or handshakes, but never initiates a transaction by itself.

In Reference 1 [USBCOMP], part b, it is explained that mass storage devices us these bulk transactions to transfer data, stating:

> Mass-storage devices use bulk transfers to exchange data. Control transfers send class-specific requests and can clear Stall conditions on bulk endpoints. For exchanging other information, a device may use either of two transport protocols: bulk only or control/bulk/interrupt (CBI). CBI is approved for use only with fullspeed floppy drives. Bulk-only is recommended for new devices of all types.

> In the bulk only protocol, a successful data transfer has 3 stages: command transport, data transport and status transport. In the command transport stage, the host sends a command in a structure called the Command Block Wrapper (CBW). In the data transport stage, the host or device sends the requested data. In the status transport stage, the device sends status information in a structure called the Command Status Wrapper (CSW).

Hence, the host apparatus always initiates the data transfer by transmitting a command, in the "command transport stage", unidirectionally to the storage medium device. Subsequently, in the "data transport stage", data for storage on the storage medium is sent from the host apparatus to the storage medium device in the case of a write command, or is sent from the storage medium device to the host apparatus in the case of a read command. Thus data for storage on the storage medium for storage on the storage medium device is transmitted bidirectionally, but this does not alter the fact that commands are only transmitted unidirectionally.

IEEE 1394 standard

For the IEEE 1394 standard, Reference 3 [FWSA] is an extract from a book concerning the IEEE 1394 standard, and Reference 4 [SBP2] is an extract from the specification of the Serial Bus Protocol 2 (SBP-2) which is the relevant standard which applies to mass storage devices connected over an IEEE 1394 bus, as noted by Reference 3 [FWSA] which states:

> A wide variety of functional devices (e.g. hard drives, video cameras, and CDROMs) can be implemented as 1394 nodes. Functional devices are termed units in the 1394 specification. Certain types of devices may have related specifications called "unit architecture" that define implementation details such as protocols, ROM entries, control and status registers, etc. Two specifications of this type are:
> • Serial Bus Protocol 2 (SBP2) Architecture – used for SCSi based mass storage functions
> • A/V unit architecture – used for audio/visual functions

Reference 4 [SBP2] explains in Section 4, Model (Informative), that in the SBP2 protocol, requests (commands) are originated from the initiators and sent to the target, in particular stating:

> Serial Bus Protocol 2 (SBP-2) is a transport protocol defined for ANSI/IEEE 1394. It defines facilities for requests (commands) originated by Serial Bus devices (initiators) to be communicated to other Serial Bus devices (targets) as well as the facilities required for the transfer of data or status associated with the commands.

Reference 4 [SBP2] explains in Sections 4.1 and 4.2 that also according to the SBP2 protocol, Serial Bus devices are targets, which are implemented as units, in particular stating:

> 4.1 Unit architecture
> In CSR architecture and Serial Bus terminology, targets implemented to this standard are units. …….

In turn, a logical unit is an instance of the device model such as a disk, and a logical unit is responsible for the execution of control or data transfer commands, Reference 4 [SBP2] stating:

> 4.2 Logical units
> A logical unit is part of the unit architecture and is an instance of a device model, e.g., disk, CD-ROM or printer. A logical unit consists of one or more device server(s) responsible to execute control or data transfer commands, one or more task sets that hold commands available for execution by the device server(s) and a logical unit number that is unique within the domain of the target.

These commands are specified in terms of requests issued by the initiator and send to the target within a data structure called an Operation Request Block and the

response to the request is indicated by the target storing data (a status block) at an address originally provided by an initiator, Reference 4 [SBP2] stating:

> 4.3 Requests and responses
> Target actions, such as a disk read that transfers data from device medium to system memory, are specified by means of requests created by the initiator and signaled to the target. The request is contained within a data structure called an operation request block (ORB). The eventual completion status of a request is indicated by means of a status block stored by the target at an address provided by the initiator.

Hence, in the context of a storage medium device, for example a hard disk, connected to a host apparatus by means of an IEEE 1394 serial bus, the host apparatus is the initiatorand the storage medium device the target. Commands (requests) are only transmitted unidirectionally to the storage medium device.

Subsequently, data for storage on the storage medium is sent from the host apparatus to the storage medium device, or is sent from the storage medium device to the host apparatus in the case of a read command. Thus data for storage on the storage medium for storage on the storage medium device is transmitted bidirectionally, but this does not alter the fact that commands are only transmitted unidirectionally.

For at least these reasons, claim 7 defines over the prior art. Limitations similar to those discussed above with respect to claim 7 are also recited by claims 1 and 13, which likewise define over the prior art. Claims 2-6, 8-12, and 14-18 depend either directly or indirectly from one of claims 1, 7, or 13 and likewise define over the prior art. Reconsideration and withdrawal of the rejections are respectfully requested.

References

Reference 1 [USBCOMP], part a:

"USB Complete" by Jan Axelson, pub Lakeview research LLC. Page numbers refer to 3rd edition 2005, 1st edition pub 1999, p 35-36.

Reference 1 [USBCOMP], part b:
"USB Complete" by Jan Axelson, pub Lakeview research LLC. Page numbers refer to 3rd edition 2005, 1st edition pub 1999, p 209-210.

Reference 2 [USB20]:
Universal Serial Bus Specification revision 2.0 April 27th 2000, section 8.5.2. Availablefrom http://www.usb.org/developers/docs/

Reference 3 [FWSA]:
Firewire System Architecture, 2nd Edition,.pub Addison Wesley 1999, p 21.

Reference 4 [SBP2]:
American National Standard for Information Technology – Serial Bus Protocol 2 (SBP-2)- ANSI NCITS 325-1998, sections 4 to 4.3.

CONCLUSION

It is believed that all of the stated grounds of rejection have been properly traversed, accommodated, or rendered moot. Applicants therefore respectfully request that the Examiner reconsider and withdraw all presently outstanding rejections. It is believed that a full and complete response has been made to the outstanding Office Action and the present application is in condition for allowance. Thus, prompt and favorable consideration of this amendment is respectfully requested.

If the Examiner believes that personal communication will expedite prosecution of this application, the Examiner is invited to telephone the undersigned at (248) 641-1600.

Respectfully submitted,

Dated: July 29, 2010

By: /MichaelMalinzak/
Michael Malinzak, Reg. 43,770

HARNESS, DICKEY & PIERCE, P.L.C.
P.O. Box 828
Bloomfield Hills, Michigan 48303
(248) 641-1600

MM/PFD/tlp

15499060.1